Claim Amendments

It is requested that the following amendments be entered into the Claims:

Claims 1 - 4 (Cancelled)

5. (Currently Amended). A second stage lottery program for a dispatcher program that dispatches tasks within an operating system of a computer system, the computer system supporting at least two classes of said tasks, each of said classes including at least two levels of said tasks, said dispatcher program to determine which of said tasks will be assigned to a next available IP resource, said second stage lottery program comprising:

a random number generator and selection program for generating a first random number for selecting one of said at least two classes, and for generating a second random number for selecting one of said at least two levels within said selected class; and

a transfer program for ~~transferring control of~~ assigning said next available IP resource ~~from said second stage lottery program~~ to execute a task assigned to said selected one of said at least two levels.

6. (Currently Amended) The second stage lottery program of claim 5 further comprising:

a level switching routine ~~for handling~~ to handle a failure by said transfer program to find a task on said selected one of said at least two levels~~, said level switching routine for selecting a different one of said at least two levels that is associated with a task, and allowing transfer of control to said~~ and to assign said next available IP resource to a task associated with ~~said~~ a different one of said at least two levels.

7. (Previously Amended) The second stage lottery program of claim 5 wherein any one of said at least two levels is two times more likely to be selected than a next lower one of said at least two levels.

8. (Currently Amended) The second stage lottery program of claim 7 wherein tasks within each level of said at least two levels ~~will have tasks of like quantum values, wherein each of said quantum values defines an amount of time a task~~ will be assigned to said next available IP resource for a same amount of time as other tasks in said level.

9. (Currently Amended). The second stage lottery program of claim 5 wherein each of said tasks ~~has~~ is respectively associated with an ~~a quantum value, said quantum value identifying a computer system specific~~ amount of time ~~in which~~ said each of said tasks ~~with said quantum value~~ may continuously execute on an ~~instruction processor~~ IP resource, and wherein said second stage lottery program employs a quantum bias routine~~, said quantum bias routine~~ comprising:

a data capture routine for determining how much of ~~an allotted segment of~~ said associated amount of time ~~quantum value a task that has executed used~~ a task executes on said IP resource before returning control to said dispatcher; and

a bias adjustment routine for adjusting said ~~allotted segment of said quantum value~~ amount of time associated with said task based on how ~~much of said allotted segment was used~~ long said task last executed on said IP resource.

10. (Currently Amended). The second stage lottery program of claim 9 wherein said bias adjustment routine does not adjust said amount of time associated with ~~current allotted segment to a new allotted segment for~~ said task if ~~use of said allotted segment~~ execution of said task was interrupted by an interrupt.

11. (Canceled)

12. (Currently Amended) A computer system having a quantum timer ~~settable~~ to allow ~~processing on~~ an IP resource ~~for a limited duration of~~ to be assigned to process one or more tasks, the computer system also having ~~an operating system having~~ a dispatcher program ~~wherein~~ whereby all said tasks are ~~identifiable~~ identified as being members of classes, ~~and wherein~~ said dispatcher program ~~comprises~~comprising:

a) a scheduler ~~code section executable~~ to determine for how long, and to which of said one or more tasks, an IP resource will be next assigned; and

b) a scheduler queue from which said one or more tasks may be assigned ~~addressable and assignable~~ to said IP resource, wherein

said scheduler ~~code section has~~ provides a two stage lottery ~~execution algorithm~~, a first stage ~~of said two stages~~ using a lottery process ~~for determining~~ to select one ~~from which class~~ of said classes ~~a next one of said one or more tasks will be selected~~, ~~said~~ a second stage ~~of said two stages~~ using a second lottery process to select from said selected class a level, wherein said scheduler selects a task that will next be assigned to said IP resource from said selected level of said selected class ~~a class determined by said first of said two stages~~.

13. (Previously Amended) The computer system set forth in claim 12 wherein said first stage chooses a class randomly from among all said classes using a bias settable by a user.

14. (Previously Amended) The computer system set forth in claim 12 wherein the number of said classes is selectable by a user.

15. (Currently Amended) The computer system as set forth in claim 12, wherein ~~if any one of said any tasks is of a~~ any tasks included within one or more predetermined classes are ~~class "above the lottery line" then said any one of said any tasks is~~ assigned

an IP resource prior to ~~running~~ said scheduler providing said two stage lottery~~code section~~.

16. (Currently Amended) The computer system as set forth in claim 12, wherein if the scheduler ~~code section's first of two stages~~ selects a class that does not have ~~which is empty of said~~ any said tasks as members, said scheduler ~~code section next~~ chooses another class~~ of available classes~~.

17. (Currently Amended) The computer system as set forth in claim 12, wherein if none of the classes has any of said ~~any~~ tasks as members, said scheduler ~~code~~ selects a ~~very~~ low priority level operating system task.

18. (Currently Amended) The computer system as set forth in claim 12, wherein said second stage lottery comprises:

a random number generator and selection program for generating a random number and for selecting ~~a one of said at least two levels~~ said selected level within said selected class based upon ~~a correspondence of~~ said ~~thereby generated~~ random number; and

a transfer program for causing said IP resource to begin execution of ~~transferring control from said second stage lottery program to a task found on~~ a task assigned to said selected level ~~one of said at least two levels~~.

19. (Currently Amended) A method for use by a dispatcher ~~algorithm in an operating system in~~ of a computer system for selecting a task to provide with an available instruction processor resource, wherein the task may be selected from one or more classes, each of which may have one or more priority levels, wherein said method comprises:

determining whether ~~within a scheduler queue~~ there are any ~~task pointers~~ tasks within said priority levels of said classes ~~of tasks on said _scheduler queue, and if so, determining whether any said any task pointers are~~ that are above a second stage lottery line, and if so assigning a first of said tasks ~~of said any tasks indicated by said any task pointers above said second stage lottery line~~ to said available instruction processor resource; but if not,

and if there is only one said priority level having any tasks, ~~of said any task pointers,~~ assigning a first of said any tasks from said only one priority level ~~corresponding to said any task pointers at said only one said priority level~~ to said available instruction processor resource; else,

running a first stage of a second stage lottery algorithm to select one of said classes, ~~and~~ running a second stage of said second stage lottery algorithm to select a priority level from said selected class, ~~one of said classes a priority level~~ and selecting a task from ~~which to select said task~~ said selected priority level to provide to said available instruction processor resource.

20. (Currently Amended) A method for use by a dispatcher ~~algorithm~~ in an operating system ~~in~~ of a computer system for selecting a task ~~from one or more classes to provide with~~ to assign to an available instruction processor resource, said task being selected from one or more classes of tasks, each having one or more priority levels, wherein said method comprises:

~~determining whether within a scheduler queue there are any task pointers within priority levels of a class of tasks on said scheduler queue, and if so, and~~

if there is only one of said priority ~~level~~ levels having any ~~of said any task pointers~~ tasks, assigning a first of said any tasks ~~corresponding to said any task pointers at said only one said priority level~~ to said available instruction processor resource, else,

running a second stage lottery algorithm, a first lottery stage of said lottery algorithm to select one of ~~the plurality of~~ said classes, and a second lottery stage of said lottery algorithm to select from said one of said classes a priority level from which to select a task to assign to said available instruction processor resource.

21. (Currently Amended) The method of claim 20 further comprising:

~~moving said task pointers within priority levels of said one or more classes on said scheduler queue, wherein each priority level can maintain a chain of said task pointers and wherein there are more than one of said priority levels, said task pointer moving process comprising:~~

~~maintaining a task assigned quantum which identifies for each task on said scheduler queue a set~~ respectively associating each task with an amount of time said task is entitled to be assigned to said instruction processor resource~~-to which said each task is entitled upon being assigned to said instruction processor resource~~;

placing ~~said~~ each task into a any one of said priority ~~level~~ levels based upon ~~a value in~~ said ~~task assigned~~ associated amount of time ~~quantum for said each task~~; and

moving a task from a current priority level to a different ~~changing said each task~~ priority level based on how much of said ~~value in said task assigned quantum for said each task said each~~ associated amount of time said task used a last time said ~~each~~ task was assigned to said available instruction processor resource.

22. (Currently Amended) The method of claim 21, ~~and~~ said moving step further comprising: ~~-determining whether said each task priority level should be changed based on said last time said each task was assigned to said available instruction processor resource whereby~~

if said task did not complete execution last time said task was assigned to said available resource, ~~did not complete within said task assigned quantum,~~ moving said task from said current priority level to a next lower ~~said task priority level for said task by one~~ priority level and increasing said associated amount of time by a predetermined amount ~~raise said value of said task assigned quantum by a factor of 2~~,

if said task used less than all, but more than a predetermined portion, of said associated amount of time last time said task was assigned to said available resource, ~~used less than said value of said task assigned quantum but more than a small portion of said task assigned quantum, leave~~ leaving said priority level and said associated amount of time ~~small portion~~ the same, and

if said task used less than said predetermined portion of said associated amount of time last time said task was assigned to said available resource, ~~used less than said small portion of said value of said task assigned quantum, increase~~ moving said task from said current priority level to a next higher ~~said~~ priority level ~~of said task by one~~ and decreasing said associated amount of time by a predetermined amount ~~halve said task assigned quantum value of said task~~.

23. (Currently Amended) The method of claim 21 wherein each task is represented by a task pointer stored on a scheduler queue ~~said task assigned quantum is assigned for said each task prior to a said task pointer being on said scheduler queue~~.

24. (Currently Amended) The method of claim 21 wherein ~~a~~ said first stage lottery ~~algorithm~~ selects which of said classes of tasks ~~on said scheduler queue will have~~ has tasks assignable to said available instruction processor resource.

25. (Currently Amended) The method of claim 21, further comprising:

if any of said tasks are in classes above a lottery line, providing said available instruction processor resource to said tasks in said classes above said lottery line in a predetermined priority order, and otherwise using ~~-wherein a~~ said first stage lottery ~~algorithm selects~~ to select which of said classes of tasks ~~on said scheduler queue will~~ have tasks assignable to said available instruction processor resource~~, but only if there are no tasks in classes above a first lottery line, and if there are tasks in said classes above said first lottery line, providing said available instruction processor resource to said tasks in classes above said first lottery line in a priority order~~.

26. (Currently Amended) A dispatcher system for use within a computer system, said computer system having an ~~available~~ instruction processor resource and a scheduler for storing pointers to tasks, each to be assigned ~~quantums of~~ to said instruction processor resource for an amount of time associated with said task, said dispatcher system comprising:

a first stage of a two stage lottery system for using a random number generator to select one of multiple classes of tasks; and

a second stage of said two stage lottery system for selecting a task from said selected one of said multiple classes, whereby said selected task is assigned to ~~an available~~ said instruction processor resource for ~~a portion of a quantum assigned to~~ said amount of time associated with said selected task.